SRPeek: Super Resolution Enabled Screen Peeking via COTS Smartphone

Abstract— **We use smartphones everywhere and privacy concerns have arisen for the "shoulder surfing" attack, where strangers peek at our phone in public areas. To mitigate this privacy threat, various countermeasures have been designed but mainly against attackers with the naked eye. With the development of smartphones and super resolution (SR) techniques, a malicious attacker can peek from further away with the assist of his/her smartphone camera and SR algorithms, rendering the underestimated threat model. To underline this concern, we present SRPeek, an end-to-end system of shoulder surfing deployed on commercial smartphones, including a unique deep neural network (DNN) architecture for multi-image SR. We implement SRPeek in Android. The experimental results demonstrate its efficiency and robustness, allowing a human to read 90% of the characters at a distance of 1.8m with traditional lenses and 6m with optical zooming lenses. And it outperforms the state-of-the-art and fills the blank space of multi-image SR for shoulder surfing attack.**

I. Introduction

Smartphones have become a necessity in our lives, and privacy concerns have consequently arisen. We often worry about nearby parties peeking at our screen, namely "shoulder surfing". Massive efforts have been made to shield users against this threat [1]–[3], but this threat itself is somewhat understudied in comparison. In our opinion, the common threat model of an attacker with his/her naked eye is outdated, as with the groundbreaking developments in smartphones and neural networks, we should have considered the possibility that the shoulder surfing attacker is looking through his smartphone's lenses and even equipped with super resolution (SR) algorithms—already in use in several commercial phones—a feature rarely considered in previous works on shoulder surfing. In our work, we'll present an up-to-date threat model as a reference for future works of screen privacy protection.

Motivation. Although it's pointed out by Eiband [4] that in most occasions the shoulder surfer means no harm and merely takes several peeks out of curiosity, we argue that a mere stranger can hardly do any harm, and it is the rare but malicious attacker, with certain familiarity and knowledge of the victim, that can utilize the stolen information and should be the primary object of our defenses. The argument goes the same with cameras—although uncommon, the utilization of this commonly available equipment can deal greater harm with longer attacking range and ability to preserve evidence. Furthermore, the vulnerable information is not limited to passwords, as although biometrics and login services can conceal passwords from screens, leakage of texts and emails can also be perilous. For example, in a Senate hearing, the Justice Secretary of the Philippines, Vitaliano Aguirre II, suffered a leakage of his text messages, as someone had taken a snapshot of his smartphone [5]. The privacy concerns have been exacerbated by the rapidly developed mobile camera module these years. Equipped with multiple cameras, the newest generation of smartphones can perform 100× zooming compared to the standard 5× of single camera phones. Hardware improvements in memory also allow the burst mode at tremendous frame rates and even high-speed photography, delivering videos with thousands of frames per second, enabling the usage of multi-frame super resolution (SR) algorithms to further improve the shoulder surfing range.

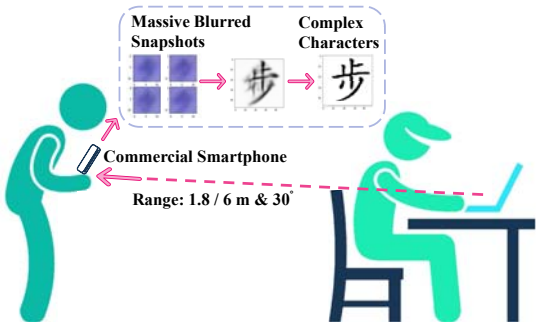


Fig. 1: Illustration of the long-range shoulder surfing threat model on smartphones, exacerbated by the SR algorithm.

Unfortunately, none of the existing works about screen privacy protection has considered the presence of modern mobile phone cameras and SR algorithms. Numerous defense mechanisms have been proposed, from physical privacy films to alternative password entry interfaces [1], [2] and input methods [3], but the additional deployment cost and interaction complexity keep them from being widely deployed [6]. Given the ubiquitous usages of smartphones and serious damages to the victim, it is imminent to recall attention on this new threat model and propose corresponding countermeasure for modern circumstances.

Threat Model. We present SRPeek, an end-to-end system of shoulder surfing using the SR algorithm, illustrated in Figure 1. While the victim's reading private information in public, the attacker, given that he/she has acquired line of sight(LoS) of the screen, uses his commercial smartphone to take 10 to 20 snapshots rapidly in burst mode (about 10 frames per second), and processes the images with multi-frame SR algorithms

to generate a high-resolution result. The whole process only takes a few seconds and can even be repeated at a 2 second interval, enabling the attacker to monitor the victim's screen continuously. Meanwhile, few people would suspect a stranger holding his smartphone at 1.8 meters behind them. Given the newest generation of smartphones equipped with telephoto lenses and capable of optical zooming, this distance can reach a staggering 6 meters, posing a silent but deadly threat to screen privacy.

Challenges. Creating a powerful SR based shoulder surfing threat model, however, is not trivial, especially for commercial smartphones in real-time. And we achieve SRPeek by overcoming four key challenges.

- **Blurriness of input images.** The most prominent one is the blurriness of captured snapshots, as they are taken secretly and hurriedly at extreme range, without especially focusing. It can be further exacerbated by straining the magnification of smartphone lenses. Unlike most SR applications and datasets working with "normally" captured snapshots (e.g., scenery, scanned images) [7], [8], these snapshots exhibit lower concentrations of information and more artifacts as Figure 1 shows, requiring a reliable "reconstruction" than "interpolation". Since blurrier input means less information, rendering a greater possibility of reconstructing the wrong character. On one hand, each snapshot is blurred with a randomly different PSF kernel, removing the consistency between neighboring frames. Thus, it cannot be approximated as video clips using a constant, isotropic Gaussian kernel; on the other hand, each defocused snapshot only contains fractions of information, analogous to pieces of a jigsaw puzzle, posing a greater demand for integration abilities of the SR algorithms than most other SR applications.
- **Super resolution with multiple snapshots.** To resolve the blurriness of input snapshots, an SR algorithm on a smartphone is required to recover the blurred contents. However, none of the existing SR algorithms can be adapted for this attack scenario, especially with massive extremely blurred snapshots as input. Since smartphones can capture 10 snapshots per second in burst mode. Specifically, the difference between videos and multiple snapshots rules out existing image/video super resolution algorithms [9], [10]. The reason has two folds. First, massive input leads to an increase in model complexity, making it unsuitable for mobile deployment. Second, the nature of our task requires to do pixel-wise correlation and comparison in the same coordinate with consecutive snapshots throughout our construction process. Otherwise, we cannot tell if, for example, it is one stroke or two parallel strokes that are shown in these two darker pixels on this snapshot.
- **Model complexity and variable input.** We have to operate our threat model locally on commercial smartphones in a real-time manner rather than remotely in cloud server. Since 1) the latter requires stable internet access, and 2) sending multiple snapshots across the Internet will consume time and bandwidth. Given the required 10 to 20 snapshots to be processed for the ideal content recovery, we have 1-2 seconds as the minimum interval between each output frame, rendering a limited period for calculation with the restricted power and RAM. In the case where the display on the target screen changes more frequently, fewer snapshots and less processing time are available for each scene. Consequently, it cannot only require a more precise network model, but also a model that can deal with dynamic inputs efficiently, either 3 or 20 snapshots.
- **Complexities of characters.** To deploy our threat model broadly for real-life scenarios (e.g., record passwords or e-mails), it is supposed to reconstruct complex characters with multiple strokes, such as Chinese characters shown in Figure 1. Composed of multiple strokes, characters are distributed discretely in the pixel-wise snapshot and different from other entities (e.g., faces and natural objects), rendering unique challenges. In some cases, messing up a stroke even slightly can shorten or lengthen a stroke, leading to a different character for misunderstandings. Unlike most SR applications working on similarity between their output and the ground truth, our network architecture and training process must be engineered to recover readable contents. Another challenge is the imbalanced element for words. Since we know certain stroke or alphabet of Chinese and English characters can occur more frequently, results in lower prediction accuracy over the less common, unconventionality shaped characters.

Summary of Evaluation Results. Equipped with our uniquely designed SR algorithm, our system can help the attacker read the characters shown on the victim's screen with above 90% accuracy at 1.8m distance on a smartphone without optical zooming(hereafter referred as traditional lens), or 6m on a phone with optical zooming(hereafter referred as optical lens) in our experimental settings. SRPeek can capture snapshots and produce high-resolution images constantly with about 2 seconds interval. Our experiments in real-life scenarios show that it can decipher texts or passwords of the victim at a safe distance, without alarming the victim, posing a serious threat to screen privacy.

Contributions. This paper makes the following contributions:

- We propose SRPeek, an end-to-end threat model of shoulder surfing from a broad range, which can be deployed on commercial smartphones. To the best of our knowledge, we are the first to consider the presence of smartphone cameras and SR algorithms in shoulder surfing scenarios.
- We design a multi-frame SR neural network architecture to reconstruct massive extremely blurred and defocused snapshots. It can be further extended to other text-recognition applications.
- We evaluate this new shoulder surfing threat model in multiple scenarios. And it outperforms the state-of-the-art for content recognition, calling for the new privacy concern.

The rest of the paper is organized as follows. Section II describes the related work, especially the state-of-the-art for shoulder surfing and SR techniques. We present the system and network design in Section III and IV, followed by the implementation in Section V and evaluation in Section VI and VII. We further wrap up this paper by delivering the limitations and countermeasures of our system in Section VIII. And the conclusion is shown in Section IX.

II. Related Work

A. Shoulder Surfing

With the arrival of the information era, privacy issues are becoming increasingly prominent. Smartphone screen privacy, the concern of our smartphones being observed by strangers in public areas, or shoulder surfing, has been studied heavily recently [4], [11], [12]. To mitigate this threat, some systems hide the information [13] or warn the user [14] once sensing malicious passers-by; others modify the user interfaces, including creating honeypots (for passwords) [15], confusing unauthorized parties [1], and making the interactions invisible [3] or unreadable from a distance [6]. Most of the works assume that the attacker is a casual passer-by, taking occasional peeks with the naked eye, as is the case most of the time [1], [4]. Given the assisted equipment, the malicious attacker can however acquire the sensitive information (passwords, business correspondence, etc.) readily to do real harm.

However, compared to the various works focusing on defenses against shoulder surfing, the works studying and modeling this threat are sparse and outdated. Most of these works focus on scenarios where the attacker peeks at the phone with his/her naked eye, performing experiments, or conducting surveys to research this threat. Eiband et al. [4] conducted a user survey to investigate stories of shoulder surfing; Kwon et al. [12] designed a shoulder surfing approach called covert attentional shoulder surfing, and evaluated its success rates against PIN entry scenarios, presenting a powerful shoulder surfing threat model with the naked eye; and Schaub et al. [16] evaluated shoulder surfing susceptibility, using Levenshtein distances and 7-point Likert scales to evaluate the accuracy and difficulties of shoulder surfing on different virtual keyboards. These works, however, lack quantitive modeling of this privacy threat, such as controlling the distances, illumination, angle, etc. to evaluate the vulnerability of screen privacy in different scenarios, and these works all focus on the unequipped attacker, so that their attack range is within 1 meter or even closer (where the attacker stands right behind or next to the victim), which is a barely practical scenario.

To deal with the threat model of tool-assisted shoulder surfing, Maggi et al. designed an automatic shoulder surfing threat model, observing the target smartphone with a digital camera [17]. However, the system contains only recognition algorithms, lacking SR processors to enhance the quality of the image, so that it can only function when the attacker is standing at close range. By tailoring and fusing the SR technology with smartphones possessing powerful lenses and processors, we propose a stronger shoulder surfing threat model in which attackers can deploy it on commercial smartphones while obtaining information for a longer range to reduce suspicion, say 1.8-6m away from the victim's screen with an observing view of 30° shown in Table I. Evaluations demonstrate its feasibility and privacy concerns in our daily life. We also performed thorough evaluations of our threat model, measuring its abilities with various environmental parameters. To the best of our knowledge, we are the first work to design and model the new form of shoulder surfing attack with the assistance of smartphones and multi-frame SR algorithms.

TABLE I: A comparison of the state-of-the-arts on shoulder surfing.

Reference	Scenarios	Metric	Quantitative Model	Distance[a]
Eiband [4]	Naked eye	×	×	-
Kwon [12]	Naked eye	✓	×	1m
Schaub [16]	Naked eye	✓	×	-[b]
Maggi [17]	Camera	✓	×	-[b]
SRPeek	COTS phones	✓	✓	1.8 / 6m

[a]The maximum distance to the victim's screen. [b]The attacker stands next to the victim without the maximum effective distance.

B. Super Resolution

Image Super Resolution is the process of reconstructing an image with a higher spatial resolution. Based on the structural patterns, self-similarity [18], or previous knowledge of the image genre, the single-image SR techniques take as input a single low-resolution image, rendering a sharp, high-resolution one by deducing missing information and reconstructing the missing pixels. Further, multi-image SR techniques work on a set of pictures on the same scene, such as multiple snapshots from a smartphone, successive images from a satellite, or adjacent frames on a video clip. These algorithms collect extra data from slight differences between these images, often exhibiting better performance than single-image SR algorithms.

Recent works on SR are mostly based on Convolutional Neural Networks(CNN) [21] and Generative Adversarial Networks(GAN) [22], the former often gets closer to ground truth while the latter generates fewer artifacts and is more pleasing to the human eye. To resolve multi-image SR tasks, say video SR, some works [10], [23] use 3-dimensional convolutions to utilize sequentiality and consistency between adjacent frames. Some works also modify the dataflow among the network layers to merge neighboring frames [24], or recurrently process the frames under the guidance of the output of the previous frame [25]. For images without consistency or sequential information, like satellite images, most works choose hybrid methods, solving the multi-image SR problem with multiple single-image SR procedures. They either merge the results of single-image SR algorithms for efficiency [26], or build a multi-image network to create a comprehensive view

based on single-image SR networks [27]. Due to the extreme blurriness of snapshots, these methods cannot deal with the new shoulder surfing threat model we proposed, which takes as input massive blurred snapshots without the consistency between frames. And they achieve limited performance, shown in Table I.

III. System Overview

We implemented a holistic system for shoulder surfing, with the neural network as the core, on a smartphone to verify the efficiency of our model. It iterates through the following steps:

Input: Under the guidance of the attacker, The smartphone will zoom in, take focus, and take several images(the number is freely adjustable) with burst mode of the target screen. As digital zooming does not put in any extra data, we'll only use maximum optical zooming(if exist) when photographing to reduce the workload of neural networks.

Alignment: The images are then aligned to mitigate tremors. Luckily, in our scenario the target is a glowing screen whose edges are easily distinguishable in most cases, and we use them for reference to align our images. The images are also spun to make the text horizontal in the process. The screen is cropped out and the rest of the image abandoned.

Adjustment: The patches differing from the background color of the screen will be carved out for processing to reduce the workload of the network. The character size is normally 10 to 20 pixels, while variations in size aren't influential as they're covered in the training data, so zooming isn't necessary.

Processing: As our SR network accepts only 9×9 patches, we will process all the 9×9 patches (with overlapping) among the input photo to generate 36×36 images (4× upscaling). The input is RGB colored, while the output, as we are not interested in color, is black and white. The three calculation processes (Alignment, Adjustment, Processing) run parallel to the input process so that the attacker can process a batch of images while the next batch is being collected simultaneously.

Output: The outputs of our SR network, the 36×36 overlapping patches, are then rearranged to their original locations and merged (using average values) into a single image. As we only carve out the texts for processing, we will then insert these high-res texts back to their original locations on one of the input images, and display the patched image to the attacker (An illustration of the patched image can be found in Fig. 4).

These steps are repeatedly executed to enable the attacker to monitor the victim at an interval of 1 to 2 seconds. If continuous monitoring is required to avoid missing the transient display, e.g. password entering, the system can simply lengthen the input phase across that period and process the data later. The workflow of our system is shown in Figure 2.

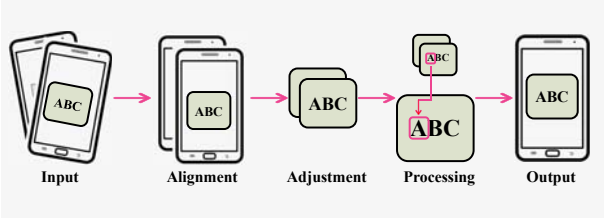


Fig. 2: Workflow of the SRPeek system.

IV. System Design

A. Design Principle

To solve the challenges and outperform the state-of-the-art for this new shoulder surfing attack, we propose a holistic system with the multi-frame SR neural network, illustrated in Figure 3.

- **Layered architecture and frequent introduction of input.** In our architecture, all input images are processed simultaneously and separately, with the information from other images for reference, throughout the network. The initial input images are introduced into the dataflow at each layer, from beginning to end, forming a model with uneven depth, acting as an anchor for the reconstruction process so that the output will be faithful to truth from beginning to end, while preserving the deep mainstream of the model to be able to process such blurry images.
- **Merging layers to adapt multiple frames.** A merging process inside each layer functions as the revenue of communication across images integrates information from all the images into one, the result of which is then stacked with each image for the next convolution. At each layer the image is exposed to the consensuses of features from other images at the same level of complexity, acting as a verification for the hypothetical features extracted from the current image, so that in the training process more audacious features can be learned and proposed without fear of punishment from the final loss, increasing the quality of featuremaps throughout the network.
- **Inconsistency problem, model complexity, and adaption for variable input.** To counter the lack of consistency between frames, all the convolutional layers throughout the network are performed individually for each image with the same set of parameters. This not only reduces parameter count and calculation power, but also avoids reliance on consistency between neighboring frames. Furthermore, this allows variations in the number of input images.
- **Compatibility with complex characters.** The discrete distribution of characters leads to the inclination of deviation and producing 'fake' results. The frequent introduction of input images is designed to mitigate this challenge. Also, instead of using Mean Square Error(MSE) as the loss function, we put a weight on each pixel before applying weighted MSE. Assuming the characters are black on white, this weight increases at darker pixels and propagates to neighboring dark pixels so that long strokes and intersections of strokes are given higher weights. This process serves as a supplement to MSE to focus more on readability and is beneficial for OCR and human reading tests.

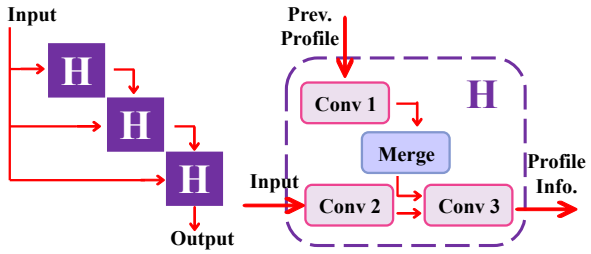


Fig. 3: Core network architecture of SRPeek.

B. Network Design

The core of the SRPeek system is a specially designed multi-frame SR neural network, accepting a group of N images indexed $x_1^{(0)}$ to $x_N^{(0)}$ as input and generating an image with higher resolution y as output. The network comprises L layers, each of which implements a non-linear transformation $H_l(\cdot)$, where l indexes the layer. As mentioned before, in each layer images are processed separately, with the merging layers as revenue for communication, so that the output of each layer is correspondent to the input. We denote the output of the l^{th} layer as $x_1^{(l)}$ to $x_N^{(l)}$, which is also the input of the $(l+1)^{th}$ layer. Till now the model is not different from traditional SR models:

$$x_i^{(l)} = H_l(x_i^{(l-1)}), i = 1, 2, ...N \tag{1}$$

Additionally, we introduce the initial images $x_i^{(0)}$ as an input for all the layers:

$$x_i^{(l)} = H_l(x_i^{(l-1)}, x_i^{(0)}), i = 1, 2, ...N \tag{2}$$

The last layer is exceptional, it yields a single image y as output.

Presently, in these layers nothing is done to raise the resolution of the images, so that the resolution of $x_i^{(0)}$ to $x_i^{(l-1)}$ and y remains the same. To increase resolution, we insert several $2\times$ nearest upsampling layers U evenly throughout the architecture between the layers:

$$x_i^{(l)} \leftarrow U(x_i^{(l)}), x_i^{(0)} \leftarrow U(x_i^{(0)}), i = 1, 2, ...N \tag{3}$$

we upsample the input images $x_i^{(0)}$ simultaneously to keep the two inputs of the following layers $H_l(x_i^{(l-1)}, x_i^{(0)})$ unanimous in resolution. For example, in a $4\times$ SR network with five layers, we may insert 2 $2\times$ nearest upsampling layers behind the 2nd and 4th layer.

Illustrated in Figure 3, inside each layer H_l there are 3 convolution layers $Conv_{1l}(\cdot)$, $Conv_{2l}(\cdot)$, $Conv_{3l}(\cdot)$ and 1 merging layer $Merge_l(\cdot)$.

Conv1: The first convolutional layer, $Conv1$, accepts the layer's first input parameter $x_i^{(l-1)}$ as input. Note that all three convolutional layers accept a single image (or its featuremaps from the last convolutional layer) as input, the convolutional process is repeated for all the images, and calculations within the same convolutional layer share the same group of parameters all the time (parameters denoted as $Param_{sl}$ for convolutional layer $Conv_{sl}, s = 1, 2, 3$):

$$a_i^{(l)} = Conv_{1l}(x_i^{(l-1)}, Param_{1l}), i = 1, 2, ...N \tag{4}$$

Merge: The results of the previous step of all the images $\{a_1^{(l)}, a_2^{(l)}, ..., a_n^{(l)}\}$ are then passed to the merging layer $Merge_l$ to generate t groups of featuremaps. Suppose the results of $Conv_{1l}$ consists of R channels:

$$a_i^{(l)} = \{a_{i1}^{(l)}, a_{i2}^{(l)}, ..., a_{iR}^{(l)}\}, i = 1, 2, ...N \tag{5}$$

The data in each channel will be merged separately in the merging layer. The output is $T \times R$ channels, denoted as $b_{tr}^{(l)} (t = 1, 2, ...T, r = 1, 2, ...R)$:

$$b_{tr(p,q)}^{(l)} = \sum_{i=1}^{N} a_{ir(p,q)}^{(l)} e^{k_t a_{ir(p,q)}^{(l)}} / \sum e^{k_t a_{ir(p,q)}^{(l)}} \tag{6}$$

where (p,q) represents the pixel at this coordinate, and k_t is a set of fixed parameters shared in all the merging layers throughout the model, controlling the behavior of the merging process. Apparently, k=0 leads to averaging, k=$+\infty$ leads to max operator and k=$-\infty$ leads to min operator. We use T=5 and k=-1,-0.5,0,0.5,1 in our model, giving consideration to both consensuses (k=0,averaging) and prominent features(k=1, 'soft'max and k=-1, 'soft'min). these $T \times R$ channels $b_{tr}^{(l)}$ is the output of this merging layer $Merge_l$.

Conv2: $Conv2$ is a replica of $Conv1$, processing the layer's second input parameter $x_i^{(0)}$, also generating N outputs with R channels per output, denoted as $c_i r^{(l)}, i = 1, 2, ...N, r = 1, 2, ...R$:

$$\begin{aligned} c_i^{(l)} &= Conv_{2l}(x_i^{(0)}, Param_{2l}), i = 1, 2, ...N \\ c_i^{(l)} &= \{c_{i1}^{(l)}, c_{i2}^{(l)}, ..., c_{iR}^{(l)}\}, i = 1, 2, ...N \end{aligned} \tag{7}$$

Conv3: The data from $Merge$ and $Conv2$ are merged together, in that all the $T \times R$ channels of $b_{tr}^{(l)}$ are replicated N times and stacked with each one of the N outputs of $Conv_{2l}$, before these N outputs, each with $(T+1) \times R$ channels, are passed through the third convolutional layer $Conv_{3l}$. There are also N output of this convolutional layer, denoted as $d_i^{(l)}, i = 1, 2, ...N$:

$$d_i^{(l)} = Conv_{3l}(Stack(c_i^{(l)}, b^{(l)}), Param_{3l}), i = 1, 2, ...N \tag{8}$$

Output: If $l < L$, this is not the last layer, the N outputs of step (4) will be the output of layer H_l. Otherwise, as we need to present a single image y as output, we add another merging and a common convolutional layer after $Conv_{3l}$. The merging layer is identical to the previous $Merge_l$, merging the N outputs $d_i^{(l)}$ into $T \times R$ channels $e_{tr}^{(l)}$, and a convolutional

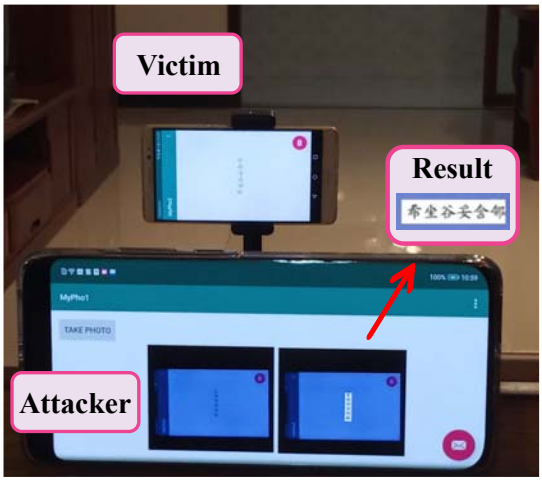


Fig. 4: Illustration of our experimental setting (distance between phones is shortened for demonstration).

layer processes this $e_{tr}^{(l)}$ to generate a single channel of output y.

$$\begin{aligned}\{e_{tr}^{(L)}, t \leq T, r \leq R\} &= Merge'_L(\{d_i^{(L)}, i \leq N\}) \\ y &= Conv'_L(\{e_{tr}^{(L)}, t \leq T, r \leq R\})\end{aligned} \tag{9}$$

V. Implementation & Training

In our experiment, we are forced to collect the training dataset on our own instead of using public datasets, as there is no publicly available image dataset built for shoulder-surfing to the best of our knowledge, and because of the uniqueness of our application, i.e. blurriness, targeting characters, working with burst mode snapshots, etc., we cannot find any publicly available substitutes.

A. Experiment Setting

Our experiment consists of 2 smartphones, one for the attacker and one for the victim. Their distance is between 1 and 2 meters for traditional lens and 5 to 7.5 meters for optical lens. Both phones are fixed to stands to keep them completely still, but the lenses with optical zoom will shift slightly from time to time, which can simulate a handheld situation. An app runs inside the attacker's phone, taking photos in burst mode repeatedly(20 images per burst), and another app runs inside the victim's phone, displaying random characters with random fonts and colors. The characters are selected from commonly used Chinese characters with 5 to 10 strokes and the English alphabet, and we divide this character assemble into training and testing subsets. As the English characters are apparently easier to classify and reconstruct for the networks, the experiments testing the performance of the network will be performed only on Chinese characters, but it is our observation that our system works on English characters as well as Chinese characters. Screenshots were taken in the victim's phone as ground truth for training and evaluation. An illustration of our experimental setting is displayed in Fig. 4.

The data was collected at different times of the day and night, with different illumination and at different positions and angles(tilting no more than 30 degrees). We collected 800,000 images in this way with a Redmi6A phone (which possesses a camera with 130 million pixels), modifying these environmental parameters every 2,000 images. This process is extremely time-consuming, but the data amount is crucial in our experiment, as slight variations of the environment will cause drastic changes in the features extracted from the characters, and only by covering the variations in each of the environmental parameters in the training data can the system successfully function in different scenarios.

B. Model Specifications

It is our observation that cameras in different phones display different patterns of distortion when performing the shoulder-surfing experiment, and apparently, images captured from a phone with weaker abilities (fewer pixels, less range of focus, worn lenses, etc.) will need a stronger, more complex model to extract the features. Thus, the specifications of our model are only for reference and may not work when reproduced on another phone. Also, we may need to repeat the data collection and training phases when switching to another phone model.

Our model accepts any number of images per burst, but it can only process a patch of 9×9 pixels(not necessarily containing whole characters, as it functions at the level of strokes) at a time. We expect that characters should be upright(the attacker can simply spin his phone). The model consists of 5 blocks described in the previous section. In each block, feature maps from the previous layer are passed through a single convolutional layer with 32 channels of feature maps as output. the 20×32 feature maps are then merged horizontally with the max-min-average process, and the output is 5×32 channels. Simultaneously, the original input images are processed again with a single convolutional layer with 32 channels output, and stacked with the former 5×32 channels to form a dataflow of 6×32 channels for each one of the images. These channels are finally passed through a single convolutional layer, outputting 32 channels per image for the next block. The kernels in each convolutional layer are 3×3. We also insert LeakyReLU and Batch Normalization processes after each convolutional layer, and 2 upsampling layers between the 5 blocks. A single 1×1 convolutional layer is placed after the 5 blocks with a single channel as output to form the final output layer. This model consists of approximately 200,000 parameters and a complexity of about 400,000 FLOPs. As a light-weighted model, it makes a prediction within 0.1 seconds on a Tesla K80 GPU over a 9×9 patch, and when implemented on a smartphone, the human user can recognize a character within 2 seconds of processing time.

VI. Model Evaluation

We perform the following experiments with two commercial off-the-shelf(COTS) smartphones: a Redmi 6A smartphone, with a single rear camera with 13 million pixels and digital

zoom only, and a HUAWEI P40 Pro, with multiple rear cameras. The telephoto camera possesses up to 5× optical zooming ability which we will utilize fully in our experiments. The Peak Signal to Noise Ratio(PSNR) metric and Optical Character Recognition (OCR) services are used to evaluate the accuracy of our system(the latter using accuracy per character).

A. Performance In Controlled Environments

In these experiments, we train and test the model with the images captured with exactly the same environment parameters. The results are shown in Fig. 5(a) and 5(b). The traditional lens group is trained and tested at 1-2 meters distance, while the optical lens, 5-7.5 meters, where less than 5% of the characters can be read with the naked eye. The PSNR and OCR results are shown in Fig. 5(a).

This model can achieve an OCR accuracy above 90% at 1.8m with traditional lens and at 6m with optical lens. Performances are relatively consistent in both day and night time, while increased distances mean less data, causing more artifacts(missing or misplaced strokes, etc.). It is the nature of Chinese characters that one mistaken stroke will largely affect its readability, leading to the result that while the pixel-wise error rises steadily with the increased distance, the accuracy will experience a drastic drop.

B. Performance In Random Environments

We train the model with data captured with varying environmental parameters and test its ability at a new environment setting. The results are shown in Figure 5(c) and 5(d). The model can achieve an OCR accuracy above 85% at 1.8m with traditional lens, and above 90% at 6m with optical lens. This verifies the efficiency of our model for environment adaption.

C. Performance with Fewer Available Images

As mentioned in sec. IV, our model is designed to work on any number of input images, which is requisite because in certain scenarios the data displayed on the victim's screen is transient and ever-changing, e.g. password entry. The previous experiments are all conducted with the ideal 20 frames per burst, but in the case study at Sec. VII and real-life scenarios the attacker only gets to photograph when the screen display remains constant, so there might be fewer images available. We evaluated the impact of fewer available images on the performance of the SR model. The results are shown in fig. 6(a) and fig. 6(b). We tested at the 1.8m daytime scenario for traditional lens and 6m daytime scenario for optical lens.

In the traditional lens group, the performance of the model drops dramatically with fewer than 10 images, indicating that not enough input information is available to rule out all possibilities of the displayed characters. The optical lens group displayed a steady descent of accuracy when fewer images are available, which is expected, as the blur patterns are more complex and the differences between frames more pronounced, so that the images are more precious to the model and contain less overlapping data, and removing any of them will cause losses in accuracy.

TABLE II: Comparison with existing systems.

System	SRPeek	SRCNN	VideoSR
PSNR	13.32db	7.69db	8.403db
OCR Accuracy	100%	10%	23%

D. Adapting Ability

We train the model with fewer groups of data, exposing it to fewer variations of a certain environment parameter when training, and examine the model's performance in random environments. The results are shown in fig. 6(c) and fig. 6(d).

We observed that variations in light and angle parameter in training data are crucial to a robust model. In the traditional lens group, the variations within 1 and 2 meters may have a smaller impact on the size of the characters so that features extracted at one distance might still exist at another. However, tilts and angles of the screen cause rotations and deformations and severely disturb the feature extraction process. The optical lens group yields similar results except at variations in distance. At a range of 6 meters, increases in distance lead to greater complexity of image blurriness, leading to more complex feature extraction and a more fragile model.

E. Comparison with Other Architectures

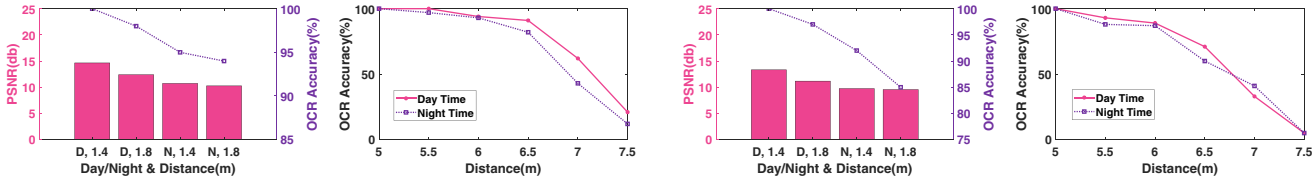
We train and test other widely used networks with the same sets of data and evaluate their results. We chose SRCNN [21], a commonly used single image SR network, applying it to every single image before merging the results by pixel-level average. We also used a multi-frame version of CNN with 3D convolutions, originally designed for video super resolution [10] (VideoSR). However, as mentioned above, it is very difficult for the single image approaches to utilize information and distinguish the noisy and deformed patterns, while VideoSR approaches rely upon consistency between frames, so they fail to give satisfactory results. We used the relatively easy 'daytime 1.2m-distance direct with traditional lens' group of data for testing. The results are shown in Table II.

We can see that the PSNR of SRPeek is 13.32dB which 73.2% and 58.6% larger than that of SRCNN and VideoSR. For the OCR accuracy, SRPeek can recognize all characters, but SRCNN and VideoSR can only recognize 10% and 23% of them. The results show the efficiency of our SR model in comparison with existing models.

VII. Case Study

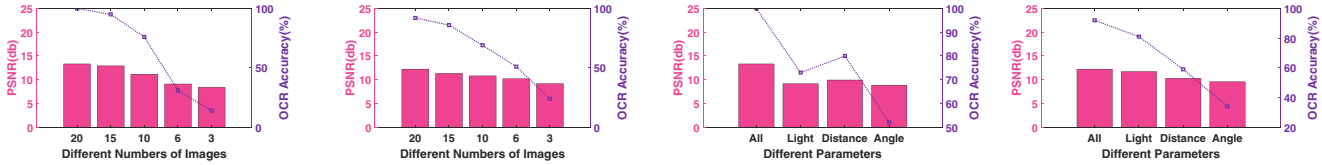
A. Accuracy

We build the system on smartphones and evaluate its performance in real-life scenarios (shown in Fig. 7). We experiment with a Redmi 6A smartphone (with a camera of 13 million pixels, no optical zooming) for the attacker and a HUAWEI



(a) Traditional Lens (b) Optical Lens (c) Traditional Lens (d) Optical Lens

Fig. 5: Performance in (a)(b). the random as well as (c)(d). the controlled environment using traditional lens and optical lens.



(a) Traditional Lens (b) Optical Lens (c) Traditional Lens (d) Optical Lens

Fig. 6: Performance (a)(b). with fewer available images and (c)(d). for the adapting ability via traditional lens and optical lens.

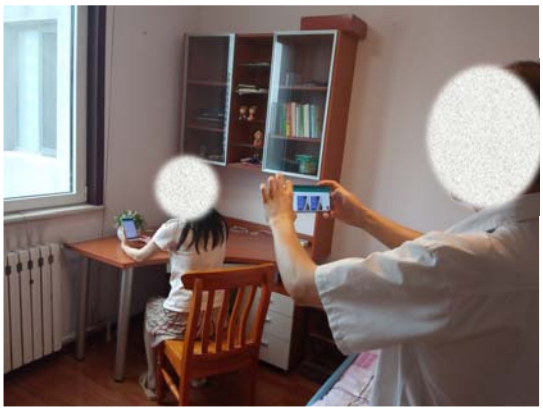


Fig. 7: Illustration of our real-life scenario (Distance between attacker and victim is shortened for demonstration).

Mate8 smartphone for the victim, with the former "1.8m daytime" setting. We ask 5 human participants to read the reconstructed characters to evaluate the usability of our model. No participants can read the unprocessed images, but all of them can decipher the information on the reconstructed image without much difficulty. The results are shown in Table III.

The results show that humans can read 95%, 85%, and 70% contents at home, transport, and theater while the OCR accuracy is 100%, 10%, and 23%. That verifies that humans can obtain the most parts of information from the peeking in various environments. In transport, the vibration of smartphone and the darker environment can fool the OCR model for content recognition in comparison with human recognition so that leads to lower accuracy in transport and theater.

B. Influence of Hand Tremors

We ask 5 participants to capture images with handheld smartphones, keeping their hands still to their greatest effort

TABLE III: Recognition accuracy in various scenarios.

Scenarios	Home	Transport	Theater
Naked Eye	5%	0	0
OCR	100%	10%	23%
Human	95%	85%	70%

TABLE IV: Impact of hand tremors when the camera (attacker phone) and/or target (victim phone) is handheld.

Accuracy	None	Camera	Target	Both
Naked Eye	5%	0	5%	5%
OCR	95%	85%	80%	80%
Human	95%	85%	80%	85%

(Handheld camera). We process these images and let them read the results. We compare this performance to the data collected on stationary phones to evaluate the influence of hand tremors. We also ask participants to hold a smartphone in their hands and read a piece of text, without other additional instructions (Handheld target). The user may freely interact with the phone when reading. We capture images of the phone at the same time to see how our system deals with a moving target screen. The results are shown in Table IV.

We can see that with the existence of tremors, the recognition accuracy drops from 95% to 85%/80% for both OCR tools and humans. We conclude from the results that hand tremors can impact the performance of our system. Hand tremors can cause motion blur and erratic shifts in the sub-pixel level(after image alignment phase) and impact performance.

C. Success rate in different tasks

We test the success rate of obtaining crucial information when the observed participant performs several tasks on a phone: reading text messages, typing text messages, entering

TABLE V: Success rate in different tasks.

Accuracy	Read text	Type text	Enter PIN	Enter password
Raw Image	5%	0	0	0
SRPeek	100%	100%	100%	80%

PIN, and typing passwords with numbers, English, and special characters(typing at 2 characters per second). We use accuracy per character as the evaluation metric. In the PIN and password tasks, fewer photos will be available, but deciphering English characters is also easier than Chinese characters, and we use specifically trained models(with the same structure and different training data). The results are shown in Table V.

We conclude that SRPeek functions normally in everyday scenarios and poses a threat to screen privacy.

D. Perceived shoulder surfing susceptibility

We ask the participants to rate the perceived shoulder-surfing susceptibility after the experiment. The attacker sits or stands at 1.8m range, pretending to be interacting with their own phone while continuously running the shoulder-surfing APP. None of the participants reported suspicion of shoulder-surfing. We believe that our system can enable a malicious attacker to gather large amounts of critical information from the victim while remaining unnoticed.

VIII. Limitations & Discussions

There are certain limitations to this work. We require a certain degree of image capturing and processing abilities of the attacker's phone, and we also expect the victim not to exert too much disturbance to the target phone.

Image Capturing Ability Latest models of smartphones can capture images at 10 frames per second in burst mode easily, however, this ability is not common in phones that are 3 years old. Heavily used phones also perform less than ideal when capturing images in burst mode.

Processing Ability To achieve the best performance the user needs a phone with strong processing capabilities to run the neural network in real-time. As neural networks have been commonplace in numerous modern APPs, most phones of the latest generation have upgraded their processing ability to run neural networks, but older versions might not possess such processing powers and cannot process images in real-time.

Motion and Tremors We assume the observed user will hold still his/her phone, and not making interactions too often. There might be extreme cases where frequent tilting of the screen may cause severe motion blur, degrading the result. Also, our work assumes LoS of the victim's screen and an angle within 30 degrees, which might not be possible if the user holds the phone too close to his body.

Although SRPeek proves to be highly efficient against unprotected screens, there are some simple methods to mitigate this unique threat while not cumbering the user.

Dynamic background. Most multi-frame SR algorithms are designed based on the assumption that all input images are reflections of the same scene, and ours is not an exception. By deploying a dynamic background behind the characters, such as tiny dots and lines traveling slowly around the screen, we can construct a constantly changing scene that will confuse the multi-frame SR algorithms, and due to the blurriness of the images, these influential elements cannot be easily removed. These dots do not need to be distinct or colored the same as the texts, as multi-frame SR algorithms function with tiny, pixel-level differences between frames, making them especially sensitive to microscopic changes.

Active scanning. There are several works providing an active countermeasure against shoulder surfing threats with the naked eye. With front-facing cameras and face detection algorithms, the smartphone can constantly scan the surrounding passers-by and detect their gaze direction, and give a warning to the user when that gaze points at the screen. However, to the extent of our knowledge, none of these works have included cameras into their detection scope, but we believe it's practical to implement such features.

Adversarial machine learning methods. In recent years we have discovered the weaknesses of neural networks and that inserting certain microscopic changes, undetectable to the human eye, to the pixels of an image will make it look different to a neural network. These methods fool the feature extraction phases of neural networks so that SRPeek is also vulnerable to this attack. Theoretically, by exerting a pattern to the victim's screen, it can be captured by the attacker's camera and confuse its SR algorithms, but these remain to be implemented in our future work.

IX. Conclusion

In this work we designed a holistic system, SRPeek, for shoulder surfing on smartphones, which serves as an up-to-date version of a threat model for shoulder surfing, and proved its efficiency. We proved that this threat towards screen privacy is imminent and can steal critical information, including personal texts or passwords, from long distances, thus escaping detection. It is our wish that this work can stir some discussion in the field of screen privacy protection and propagate defense mechanisms across critical mobile apps.

The core of SRPeek is a specially designed multi-frame SR network. With its innovative architecture, this network outperforms other algorithms of the same field in our application. The design ideology enables this network to process higher levels of data integration ability while keeping a low calculation profile, and we believe the elements of this design can be used in other applications with large amounts of data, such as natural language processing or anomaly detection. Our model can also be used in OCR tasks when multiple images are available, functioning as a preprocessor to improve the quality of the images and increase accuracy.

REFERENCES

[1] S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget, "Design and evaluation of a shoulder-surfing resistant graphical password scheme," in *Proceedings of the working conference on Advanced visual interfaces*, 2006, pp. 177–184.

[2] A. Papadopoulos, T. Nguyen, E. Durmus, and N. Memon, "Illusionpin: Shoulder-surfing resistant authentication using hybrid images," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 2875–2889, 2017.

[3] M. Kumar, T. Garfinkel, D. Boneh, and T. Winograd, "Reducing shoulder-surfing by using gaze-based password entry," in *Proceedings of the 3rd symposium on Usable privacy and security*, 2007, pp. 13–19.

[4] M. Eiband, M. Khamis, E. Von Zezschwitz, H. Hussmann, and F. Alt, "Understanding shoulder surfing in the wild: Stories from users and observers," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 4254–4265.

[5] Polotiko, "Aguirre furious at photo leak of private text message," 2017. [Online]. Available: https://politics.com.ph/aguirre-furious-photo-leak-private-text-message/

[6] C.-Y. D. Chen, B.-Y. Lin, J. Wang, and K. G. Shin, "Keep others from peeking at your mobile device screen!" in *The 25th Annual International Conference*, 2019.

[7] H. Nasrollahi, K. Farajzadeh, V. Hosseini, E. Zarezadeh, and M. Abdollahzadeh, "Deep artifact-free residual network for single-image super-resolution," *Signal, Image and Video Processing*, vol. 14, no. 2, pp. 407–415, 2020.

[8] J. Lyn and S. Yan, "Image super-resolution reconstruction based on attention mechanism and feature fusion," *arXiv preprint arXiv:2004.03939*, 2020.

[9] A. Lucas, S. Lopez-Tapia, R. Molina, and A. K. Katsaggelos, "Generative adversarial networks and perceptual losses for video super-resolution," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3312–3327, 2019.

[10] A. Kappeler, S. Yoo, Q. Dai, and A. K. Katsaggelos, "Video super-resolution with convolutional neural networks," *IEEE Transactions on Computational Imaging*, vol. 2, no. 2, pp. 109–122, 2016.

[11] W. Goucher, "Look behind you: the dangers of shoulder surfing," *Computer Fraud & Security*, vol. 2011, no. 11, pp. 17–20, 2011.

[12] T. Kwon, S. Shin, and S. Na, "Covert attentional shoulder surfing: Human adversaries are more powerful than expected," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 6, pp. 716–727, 2013.

[13] F. Brudy, D. Ledo, and S. Greenberg, "Is anyone looking? mediating shoulder surfing on public displays (the video)," in *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, 2014, pp. 159–160.

[20] G. Y. Youm, S. H. Bae, and M. Kim, "Image super-resolution based on convolution neural networks using multi-channel input," in *Proceedings of the IEEE Image, Video, & Multidimensional Signal Processing Workshop*, 2016.

[14] A. Saad, M. Chukwu, and S. Schneegass, "Communicating shoulder surfing attacks to users," in *Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia*, 2018, pp. 147–152.

[15] N. Chakraborty and S. Mondal, "Tag digit based honeypot to detect shoulder surfing attack," in *International Symposium on Security in Computing and Communication*. Springer, 2014, pp. 101–110.

[16] F. Schaub, R. Deyhle, and M. Weber, "Password entry usability and shoulder surfing susceptibility on different smartphone platforms," in *Proceedings of the 11th international conference on mobile and ubiquitous multimedia*, 2012, pp. 1–10.

[17] F. Maggi, A. Volpatto, S. Gasparini, G. Boracchi, and S. Zanero, "Poster: Fast, automatic iphone shoulder surfing," in *Proceedings of the 18th ACM conference on Computer and communications security*, 2011, pp. 805–808.

[18] N. Suetake, M. Sakano, and E. Uchino, "Image super-resolution based on local self-similarity," *Optical review*, vol. 15, no. 1, pp. 26–30, 2008.

[19] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar, "Robust shift and add approach to superresolution," in *Applications of Digital Image Processing XXVI*, vol. 5203. International Society for Optics and Photonics, 2003, pp. 121–130.

[21] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[22] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[23] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1874–1883.

[24] Y. Huang, W. Wang, and L. Wang, "Video super-resolution via bidirectional recurrent convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 1015–1028, 2017.

[25] M. S. Sajjadi, R. Vemulapalli, and M. Brown, "Frame-recurrent video super-resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6626–6634.

[26] M. Kawulok, P. Benecki, S. Piechaczek, K. Hrynczenko, D. Kostrzewa, and J. Nalepa, "Deep learning for multiple-image super-resolution," *IEEE Geoscience and Remote Sensing Letters*, 2019.

[27] Z. Dong, S. Zhang, B. Ma, D. Qi, L. Luo, and M. Zhou, "A hybrid multi-frame super-resolution algorithm using multi-channel memristive pulse coupled neural network and sparse coding," in *Proceedings of the 7th International Conference on Information, Communication and Networks (ICICN)*, 2019.